1 Makeham's User Guide

This guide provides important details regarding the implementation of Makeham's Law in R.

1.1 Overriding Functions

Functions and variables used by the `makehams` package can be over-ridden using the `gl.a(var, val)` function and be accessed using `gl.g(var)` where var is the object name and val is value to be assigned. Additionally, all defined variables in the environment can be listed using the following technique.

```
> library(makehams)
> ls(envir=gl)

[1] "A"     "B"     "c"     "d"     "i"     "radix" "uxt"   "w"     "x"
```

1.2 Select Period

An important implementation detail is regarding the select period. In all cases, a select period is assumed by default when using an actuarial function. This means that

- ${}_tp_{[x]}$ is implemented such that using a non-zero s arugment results in ${}_tp_{[x]+s}$
- $A_{[x]}$ is implemented such that using a non-zero s arugment results in $A_{[x]+s}$
- $\mu_{[x]}$ is implemented such that using a non-zero s argument results in $\mu_{[x]+s}$

  $\vdots$

For instance, lets say that the select period is 2 and the value of A_{20} is wanted. Calling `Ax(20,s=2)` actually gives the value $A_{[20]+2}$ and not A_{20}. Therefore, this value would have to be calculated as $A_{[18]+2}$ which is `Ax(18,s=2)`

To generalize the model to any select period, numerical integration was used in the implementation of several functions. For instance, the functions implementing ${}_tp_x$ and A_x use numerical integration. Although this provides for flexibility in changing the model parameters, the disadvantages of such an approach are

- Running code such as building life tables takes noticeably longer when a large select period is used, such as $d = 10$
- In addition to a function using numerical integration potentially being slow, it is also less accurate than solving an integral before programming the function

1.3 Optional arguments

Typically, as is the case with the $A_{[x]}$ function, rather than implementing new functions such as $\bar{A}_{[x]}$, these are optional parameters to the existing function. For instance, $\bar{A}_{[x]}$ can be calculate as `Ax(x,c=1)` where c is an optional parameter indicating that a continuous expected present value should be calculated.

```
> library(makehams)
> head(createLifeTable(x=20))
   x   l[x]+0   l[x]+1      lx+2 x+2
1 NA       NA       NA 100000.00  20
2 NA       NA       NA  99975.04  21
3 20 99995.08 99973.75  99949.71  22
4 21 99970.04 99948.40  99923.98  23
5 22 99944.63 99922.65  99897.79  24
6 23 99918.81 99896.43  99871.08  25
```

Another table that can be readily accessed is the insurance table.

```
> head(createInsuranceTable(x=20))
   x       A[x]     A[x]+1       Ax+2     5E[x]   5E[x]+1      5Ex+2 x+2
1 20 0.04917546 0.05143193 0.05377599 0.7825547 0.7825077 0.7824769  22
2 21 0.05139908 0.05376425 0.05622182 0.7825368 0.7824872 0.7824536  23
3 22 0.05373095 0.05620990 0.05878622 0.7825168 0.7824641 0.7824275  24
4 23 0.05617607 0.05877410 0.06147464 0.7824942 0.7824381 0.7823980  25
5 24 0.05873967 0.06146230 0.06429274 0.7824688 0.7824089 0.7823650  26
6 25 0.06142720 0.06428015 0.06724641 0.7824403 0.7823761 0.7823278  27
```

1.4 Recursions

The following recursion relationships hold

$$\begin{aligned}
A_{[x]+d} &= A_{x+d} \\
A_{[x]+d-1} &= q_{[x]+d-1}v + p_{[x]+d-1}v(A_{x+d}) \\
A_{[x]+d-2} &= q_{[x]+d-2}v + p_{[x]+d-2}v(A_{[x]+d-1}) \\
&\vdots \\
A_{[x]} &= q_{[x]}v + p_{[x]}v(A_{[x]+1})
\end{aligned}$$

where A_{x+d} can be calculated recursively using

$$A_x = vq_x + vp_xA_{x+1} \tag{1}$$

Therefore the approach is to

- Calculate A_{x+d} for $x = \omega - d - 1$ to $x - d$
- Calculate $A_{[x]+d-t}$ for $t = 1$ to d using $x = \omega - d - 1$ to $x - d$

1.5 Extension to Multiple Decrement Models

Consider the following example of a pension service table taken from `Actuarial Mathematics for Life Contingent Risks, 2nd edition`.

Example 10.5

Using the following forces of decrement, build a pension plan service table for ages $x = 20$ to $x = 65$

$$\mu_x^{01} = \begin{cases} 0.1 & x < 35 \\ 0.05 & 35 \leq x < 45 \\ 0.02 & 45 \leq x < 60 \\ 0 & x \geq 60 \end{cases}$$
$$\mu_x^{02} = \begin{cases} 0.01 \end{cases}$$
$$\mu_x^{03} = \begin{cases} 0 & x < 60 \\ 0.1 & 60 < x < 65 \end{cases}$$
$$\mu_x^{04} = \begin{cases} A + Bc^x \end{cases}$$
$$\mu_x^{(\tau)} = \sum_{j=1}^{4} \mu_x^{(j)}$$

This could be implemented using the functions already built into makehams in the following way

```
> gl.a(uxt01, function(t,x=gl.g(x)) ifelse(x+t<35,0.1,
+          ifelse(x+t<45,0.05,ifelse(x+t<60,0.02,0))))
> gl.a(uxt02, function(t,x=gl.g(x)) t^0*0.001)
> gl.a(uxt03, function(t,x=gl.g(x))
+        ifelse(x+t<60,0,ifelse(x+t<65,0.1,0)))
> gl.a(uxt04, function(t,x=gl.g(x), A=gl.g(A), B=gl.g(B),
+          c=gl.g(c)) A + B*c^(x+t))
> gl.a(uxt, function(t,x=gl.g(x),...) gl.g(uxt01)(t,x) +
+         gl.g(uxt02)(t,x) + gl.g(uxt03)(t,x) + gl.g(uxt04)(t,x))
> gl.a(radix,1e+06)
> tpxij <- function(t,x=gl.g(x),uxt=gl.g(uxt)) {
+   integrate(function(s) tpx(s,x)*uxt(s,x), 0, t)$value
+ }
```

Now that we have implemented the forces of decrement, we can build the service table

```
> p = tpx(0:45,20)
> st = data.frame(x = 20:65, lx=gl.g(radix)*p, wx=p*gl.g(radix)*
+            sapply(0:45, function(k) tpxij(1,20+k,uxt=gl.g(uxt01))),
+                ix=p*gl.g(radix)*sapply(0:45, function(k) tpxij(1,20+k,
+                  uxt=gl.g(uxt02))), rx=p*gl.g(radix)*sapply(0:45, function(k)
+                    tpxij(1,20+k,uxt=gl.g(uxt03))), dx=p*gl.g(radix)*sapply(0:45,
+                      function(k) tpxij(1,20+k,uxt=gl.g(uxt04))))
> st[41:46,] = st[41:46,]*0.7
```

```
> st[46,]$wx = 0; st[46,]$dx = 0; st[46,]$ix = 0
> st[46,]$rx = st[46,]$lx
> print(st)
      x         lx        wx        ix       rx        dx
1  20.0 1000000.00 95104.164 951.04164    0.000 237.41893
2  21.0  903707.38 85946.182 859.46182    0.000 217.71579
3  22.0  816684.02 77669.758 776.69758    0.000 199.95897
4  23.0  738037.60 70190.027 701.90027    0.000 183.96186
5  24.0  666961.71 63430.295 634.30295    0.000 169.55582
6  25.0  602727.56 57321.250 573.21250    0.000 156.58845
7  26.0  544676.51 51800.251 518.00251    0.000 144.92203
8  27.0  492213.33 46810.690 468.10690    0.000 134.43210
9  28.0  444800.10 42301.406 423.01406    0.000 125.00620
10 29.0  401950.68 38226.165 382.26165    0.000 116.54268
11 30.0  363225.71 34543.182 345.43182    0.000 108.94969
12 31.0  328228.15 31214.695 312.14695    0.000 102.14422
13 32.0  296599.16 28206.579 282.06579    0.000  96.05126
14 33.0  268014.46 25487.990 254.87990    0.000  90.60300
15 34.0  242180.99 23031.058 230.31058    0.000  85.73817
16 35.0  218833.88 10665.313 213.30626    0.000  83.45331
17 36.0  207871.81 10130.950 202.61899    0.000  83.57433
18 37.0  197459.10  9623.359 192.46717    0.000  83.98048
19 38.0  187555.08  9140.558 182.81115    0.000  84.67124
20 39.0  178154.16  8682.273 173.64546    0.000  85.66180
21 40.0  169205.82  8246.044 164.92087    0.000  86.94715
22 41.0  160707.91  7831.763 156.63526    0.000  88.54569
23 42.0  152630.97  7437.995 148.75990    0.000  90.46319
24 43.0  144955.22  7063.776 141.27553    0.000  92.71087
25 44.0  137656.06  6707.906 134.15813    0.000  95.29724
26 45.0  130718.70  2586.134 129.30670    0.000  99.73394
27 46.0  127904.96  2530.383 126.51913    0.000 106.23280
28 47.0  125139.49  2475.580 123.77898    0.000 113.44283
29 48.0  122427.60  2421.829 121.09146    0.000 121.43776
30 49.0  119795.00  2369.639 118.48197    0.000 130.32231
31 50.0  117145.49  2317.107 115.85533    0.000 140.07423
32 51.0  114571.66  2266.061 113.30306    0.000 150.88393
33 52.0  112042.19  2215.883 110.79413    0.000 162.81549
34 53.0  109552.57  2166.481 108.32404    0.000 175.96933
35 54.0  107101.92  2117.837 105.89187    0.000 190.45958
36 55.0  104687.73  2069.901 103.49506    0.000 206.40736
37 56.0  102307.56  2022.623 101.13113    0.000 223.94344
38 57.0   99945.13  1975.679  98.78394    0.000 243.17498
39 58.0   97644.13  1929.931  96.49657    0.000 264.36658
40 59.0   95353.07  1884.361  94.21806    0.000 287.56060
41 42.0   65159.77     0.000  61.87556 6187.556 210.42408
42 42.7   58702.66     0.000  55.73331 5573.331 211.51701
```

43	43.4	52859.03	0.000	50.17455	5017.455	212.66312
44	44.1	47579.08	0.000	45.15190	4515.190	213.87250
45	44.8	42805.99	0.000	40.61134	4061.134	215.10935
46	45.5	38488.26	0.000	0.00000	38488.265	0.00000